Assignment 01: Building an EV database

Barry Denby

February 17, 2022

1 Assignment Information

Course:	MSCC/MSCBD
Stage / Year:	1
Module:	Cloud Computing
Semester:	2
Assignment:	1 of 3
Date of Issue:	2022-02-17
Assignment Deadline:	2022-04-03 @ 23:55 (End of week 7)
Assignment Submission:	Upload to Moodle
Assignment Weighting:	10% of Module

2 Introduction

NOTE: read the whole assignment brief first before implementing it contains very important information

In this assignment you will be tasked with building your first PaaS application which will be a database or Electric Vehicles. You will be required to deal with users who login and logout and alter the functionality for each.

In the application users should be able to add and update cars that are in the database. They should also be able to perform comparisons of multiple cars in the database.

Be aware that the main issue you will run into here is the use of the NoSQL database. You will be fairly limited on the kind of queries you can run against the database, thus you will need to think very carefully about how you structure and store your data to make it easy to query.

NOTE: This is an individual assignment. It is not a group assignment. You can discuss ideas/algorithms but you cannot share code/documentation

3 Submission and Penalties

You are required to submit two separate components to the Moodle

- An archive containing your complete Google App Engine Python project. The accepted archive formats are: zip, rar, 7z, tar.gz, tar.bz2, tar.xz. The use of any other archive format will incur a 10% penalty before grading.
- A PDF containing documentation of your code. **If you do not provide documentation your code will not be marked.** Copying and pasting code into a PDF does not count as documentation.

There are also a few penalties you should be aware of

- Code that fails to compile will incur a 30% penalty before grading. At this stage you have zero excuse to produce non compiling code. I should be able to open your project and be able to compile and run without having to fix syntax errors.
- The use of libraries outside the SDK will incur a 20% penalty before grading. You have all you need in the standard SDK. I shouldn't have to figure out how to install and use an external library to get your app to work
- **An omission of a git repository attached to your email address that is registered for GCD will result in your application and documentation not being graded.**
- The standard late penalties will also apply

You are also required to submit as part of your archive a working Git repository.

- When I unpack your archive there should be a .git directory as part of it.
- This should be a fully working **local** git archive. It should not require access to a remote repository
- You are not permitted to upload your work to Github, Gitlab, or any other publicly visible git repository (assignment will be marked as a zero if it is)
- If you need a remote git repository the only permitted one is the college provided Gitlab which can be found at gitlab.griffith.ie
- There must be a minimum of seven commits in the git repository, one per completed bracket.

Very Important: Take note of the groups listed below. These are meant to be completed in order. Groups must be completed in full before the next group will be evaluated. Completed will mean that all tasks in the groups are visible and testable. If a single one is not visible and testable further groups will not be considered. e.g. if there are four tasks in Group 1 and task 3 is skipped or not visible or testable then Groups 2, 3 and 4 will be ignored. Documentation

will be treated separately irrespective of how many Groups you have completed.

You should also be aware that I will remove marks for the presence of bugs anywhere in the code and this will incur a deduction of between 1% and 15% depending on the severity. If you have enough of these bugs it is entirely possible that you may not score very many marks overall. I want robust bug free code that also validates all user input to make sure it is sensible in nature. Please be aware of the major bugs section. If any of these bugs are present in your application you will lose 20% for each one up to a maximum of 60%

4 Plagiarism

Be aware that we take plagiarism very seriously here. Plagiarism is where you take someone else's work and submit it as if it was your own work. There are many different ways plagiarism can happen. I will list a few here (this is not exhaustive):

- Finding something similar online (full implementation or tutorial) that does the same job and submit that.
- Finding something similar online (full implementation or tutorial) and transcribing (i.e. copying it out by hand)
- Working together on an individual assignment and sharing code together such that all implementation look the same.
- Getting a copy of someone else's code and submitting/transcribing that
- Paying someone to do your assignment
- Logging into someone elses Moodle account, downloading their assignment and uploading it to your own Moodle account.

I've had to deal with many cases of plagiarism over the last six years so I can spot it and diagnose it easily, so don't do it. To prevent plagiarism include but not limited to the following:

- Do all your code by yourself
- Don't share your code with anyone, particularly if anyone looks for a copy of your code for reference.
- Don't post your code publicly online. Remember the use of GitHub, Gitlab, BitBucket etc is prohibited.
- If you need to find information online only query about very specific problems you have don't look for a full assignment or howto.
- Change the default password on your Moodle account. The default password can be determined if someone is connected to you through social media or they get one or two details from you.

Be aware that if you submit your assignment you accept that you understand what plagiarism is and that your assignment is not plagiarised in any way.

5 Coding Tasks (80%)

- Group 1 tasks (20%)

  1. Application that has a working login/logout service.
  2. Datastore object to represent an EV it should have attributes to represent.
     - name, manufacturer, year, battery size (Kwh), WLTP range (Km), cost, power (Kw).
  3. Seperate page for adding an EV to the database.
  4. Add in a form for querying the EV database. it should be able to filter on all attributes.
     - Attributes that are string based should have a single input string.
     - Attributes that are numerical should specify a range with upper and lower limit (limits included in search).
     - This should be accessible regardless of login status.
     - Display a list of EVs that satisfy the query as a list of hyperlinks.
     - If nothing selected on the form then show full EV list

- Group 2 tasks (40%)

  5. When a hyperlink has been clicked go to a seperate page showing the information of that EV.
  6. Add the ability to edit an EV from its information page.
  7. Add the ability to delete an EV from its information page.
  8. Seperate page for comparing 2 or more EVs.

- Group 3 tasks (60%)

  9. On EV comparison page for each attribute highest value should be highlighted green and lowest red. Opposite for cost.
  10. On EV comparison page EV names should be hyperlinks to information page.
  11. Add in the ability for a logged in user to submit a review of an EV including a text field (limited to 1,000 characters) and a rating out of 10.
  12. The reviews should be visible to all users even if logged out

- Group 4 tasks (80%)

  13. On the information page for an EV this should show the average score of all reviews.
  14. On the information page for an EV the reviews should be shown in reverse chronological order.
  15. On the comparison page for EVs the average score should be added in and highlighted green if highest and red if lowest.
  16. UI Design: well thoughout UI design that is intuitive and easy to use.

- Major bugs (presence of any one of these will be a 20% reduction in mark up to a maximum of 60%)
  - An EV with same manufacturer, name, and year can be added.
  - Page for adding an EV can be accessed and used by a non-logged in user.
  - Deletion of an EV deletes the wrong EV
  - Editing of an EV can be accessed and performed by a non logged in user
  - Deletion of an EV can be done by a non logged in user
  - Comparison can be triggered if 0 or 1 EV is selected

6 Documentation Brackets (20%)

NOTE: Documentation should be around 1,500 words in length total

1. (0 to 10%): Document every method in your code from a high level perspective. i.e. give an overview of what the method does. Do not copy and paste code you will be penalised for this.
2. (10 to 20%): Document every datastructure and model you have used in your code and why you chose them.